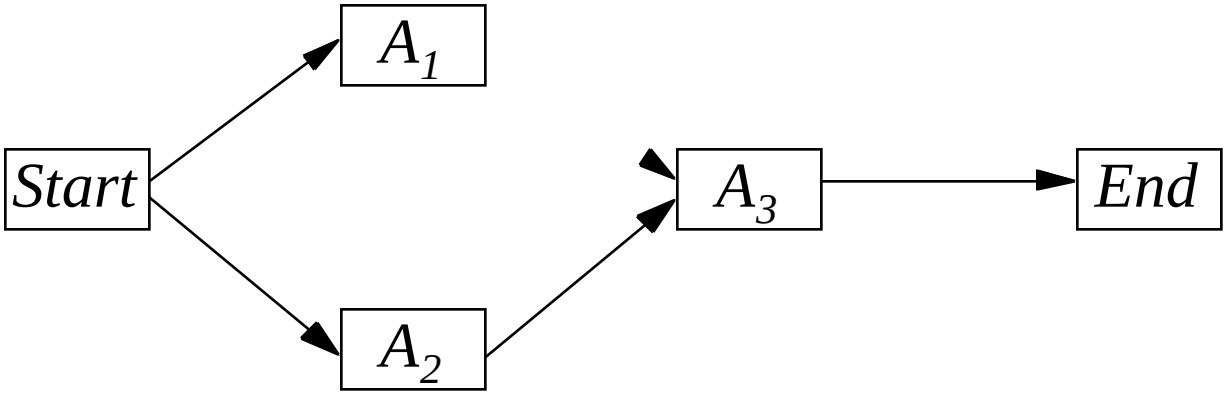
Search through plan space

- Planning as a search through plan space
  - nodes represent partially specified plans
  - edges denote plan-refinement operations
- Facilitates thinking about alternative
  - representations of partially specified plans
  - plan-refinement operators
- Partial order plan representations
  - a plan is represented as a partially ordered sequence
  - planning algorithm practices *least commitment*

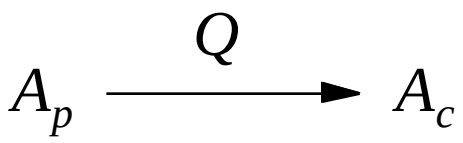
Partial order plans

- A plan is a three tuple <*A*, *O*, *L*>
  - *A* is the set of *actions* in the plan
    - initially we consider only ground actions
  - *O* is the set of *ordering constraints* among the actions
    - *O* must be consistent
  - *L* is the set of *causal links*



Causal links

- Records *dependencies* between actions
- A causal link consists of
  - the producer A_p
  - the consumer A_c
  - the proposition Q
- A causal link states that
  - action A_p has an effect Q
  - action A_c has a precondition Q
  - Q will be made true for A_c by A_p



Threats

- An action A_t is a *threat* to a causal link $A_p \xrightarrow{Q} A_c$ in a plan $<A, O, L>$ if
  - $O \quad \{A_p < A_t < A_c\}$ is consistent
  - A_t has $\neg Q$ as an effect

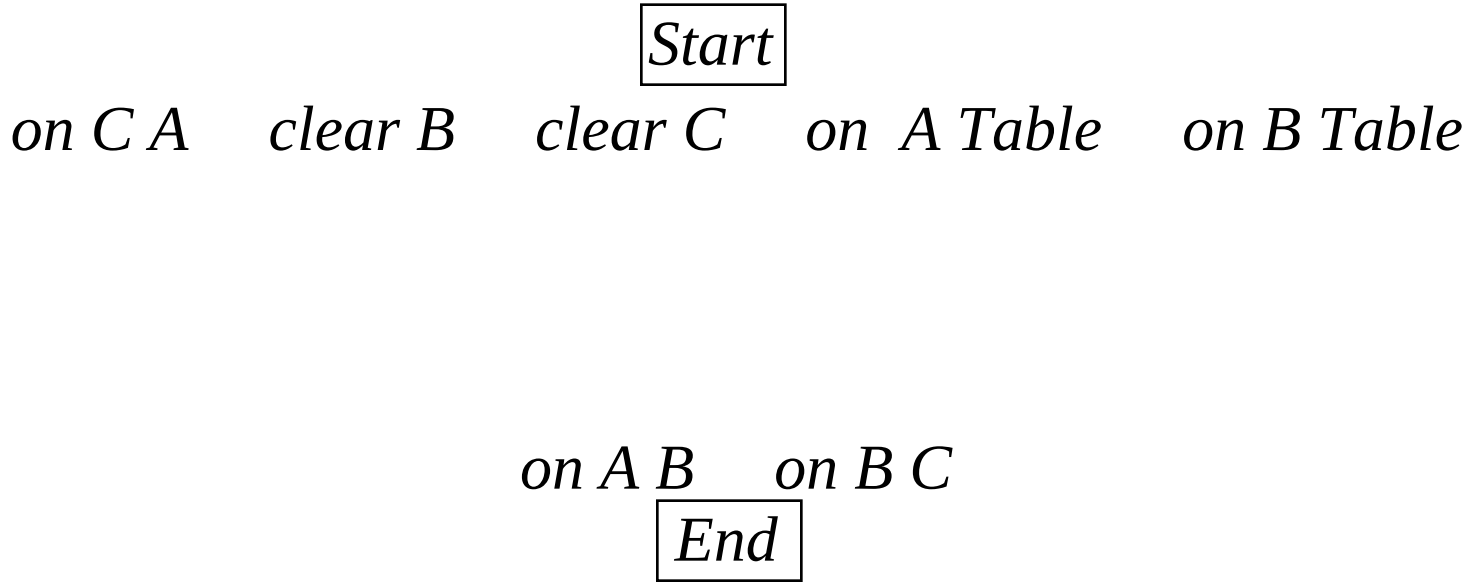
Resolving a threat

- Threats are resolved by adding additional ordering constraints
- *Promotion*
  - add the constraint $A_c < A_t$
- *Demotion*
  - add the constraint $A_t < A_p$

Representing null plans

Introduce two pseudo-actions

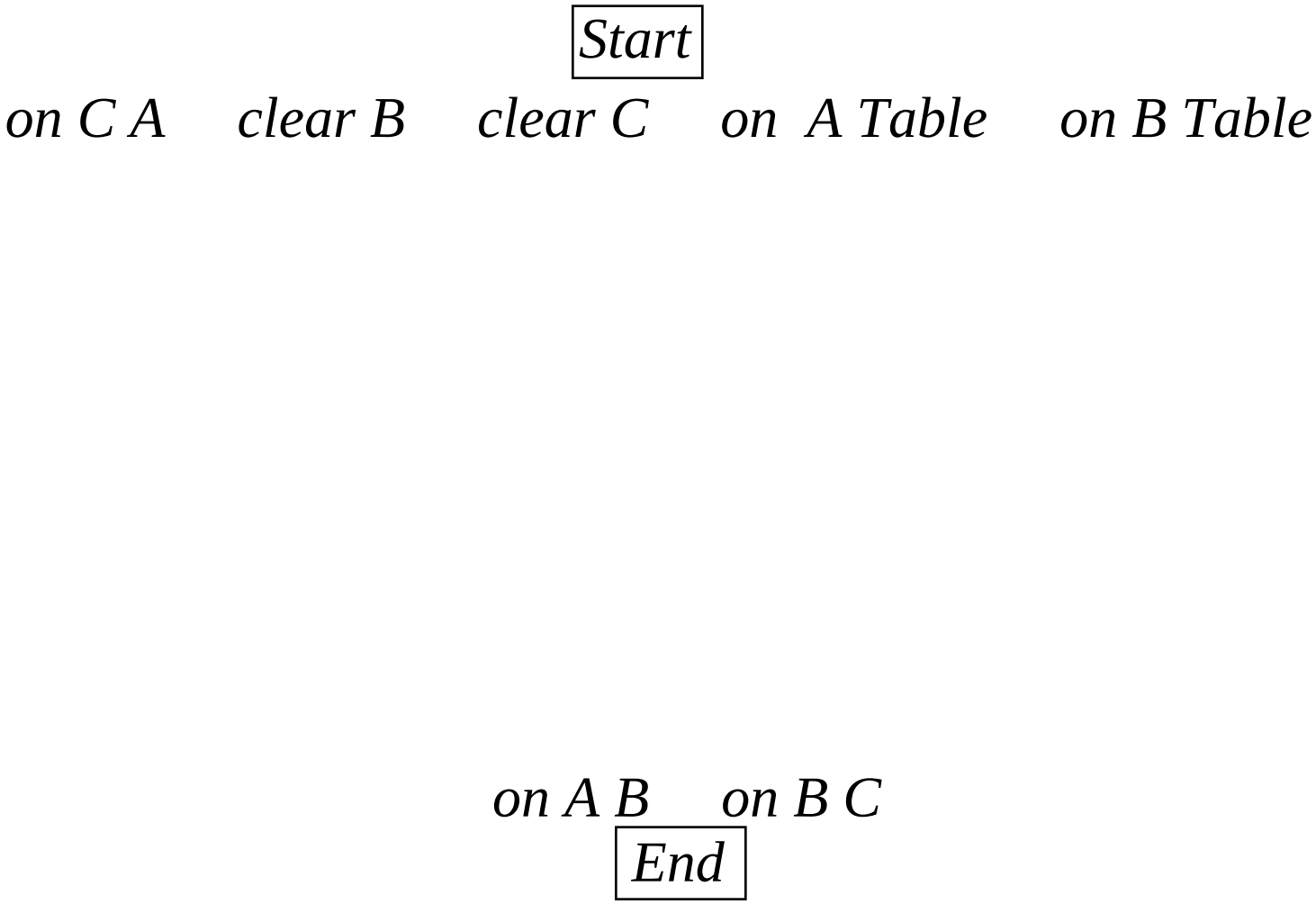
- *Start*: effects are the initial state and preconditions are empty
- *End*: preconditions are the goals and effects are empty
- Ordering constraint: *Start* < *End*



Partial order planning

algorithm *POP*(<*A*, *O*, *L*>, *Agenda*)
    if *Agenda* is empty **then return** <*A*, *O*, *L*>
    Remove some <*Q*, A_{need}> from *Agenda*
    A_{add} = **choose** an action that can be ordered before A_{add}
        and has effect *Q* (an existing or newly instantiated action)
    if no such A_{add} exists **then return** "no plan"
    Add $A_{add} \xrightarrow{Q} A_{need}$ to *L* and $A_{add} < A_{need}$ to *O*
    if A_{add} is new **then**
        $A = A \quad \{A_{add}\}$, add *preconds*(A_{add}) to *Agenda* and *Start* < A_{add} < *End* to *O*
    for every causal link $A_p \xrightarrow{R} A_c$ threatened by some A_t **do**
        choose either to *demote* A_t or to *promote* A_t consistently
        if neither ordering is consistent **then return** "no plan"
    endfor
    POP(<*A*, *O*, *L*>, *Agenda*)
end *POP*

Example



A complete plan

- Figure 10 from Weld 1994

Lifting

- Use *least-commitment* in operator instantiation
- An effect can support a precondition if the two *unify*
  - add *co-designation* constraints resulting from the *most general unifier* of the effect and precondition
  - instantiate an operator by creating a copy with a fresh set of variables
- Operator preconditions can be non-codesignation constraints
- Threats need be resolved only when they are forced
  - means that a plan is complete only when all variables are instantiated
  - ensure by requiring that variables in preconditions and effects are the same and initial state has no variables

Lifted *POP*

algorithm *LPOP*(<*A*, *O*, *L*, *B*>, *Agenda*)
  if *Agenda* is empty **then return** <*A*, *O*, *L*>
  Remove some <*Q*, A_{need}> from *Agenda*
  A_{add} = **choose** an action that can be ordered before A_{add}
    and has an effect that consistently unifies with *Q* (existing or new action)
  if no such A_{add} exists **then return** "no plan"
  Add $A_{add} \xrightarrow{Q} A_{need}$ to *L*, $A_{add} < A_{need}$ to *O*, and codesignation constraints from unification to *B*
  if A_{add} is new **then**
    $A = A \cup \{A_{add}\}$ and $Start < A_{add} < End$ to *O*
    Add noncodesignation preconds of A_{add} to *B* and logical ones to *Agenda*
  endif
  for every causal link $A_p \xrightarrow{R} A_c$ threatened by some A_t **do**
    choose either to *demote* A_t or to *promote* A_t consistently
    if neither ordering is consistent **then return** "no plan"
  endfor
  LPOP(<*A*, *O*, *L*, *B*>, *Agenda*)
end *LPOP*

UCPOP

- Augments the basic STRIPS operator representation with
  - conditional effects
  - disjunctive preconditions
  - universal quantification